# Math 151A: Problem Set 3

**Owen Jones**

5/1/2023

**Problem 1: (C) Newton's Method**

Use Newton's method to find solutions accurate to within $10^{-5}$ for the problem:

$$e^x + 2^{-x} + 2\cos x - 6 = 0 \text{ for } 1 \leq x \leq 2.$$

Repeat using the Secant method. Report the number of iterations needed to reach your computed solutions.

*For this problem: The true root is* $p = 1.82938360193385$*, you can use the stopping criterion:* $|p_n - p| \leq 10^{-5}$*.*

**Solution:**

$f \in C^2[1,2]$, $p \in (1,2)$, $f(p) = 0$ $f'(p) \neq 0$, so there exists a sequence $(p_n)_{n=1}^{\infty}$ that converges to $p$ for any initial $p_0$ within some $\delta$ of $p$. Let $p_0 = 1$, (and $p_1 = 1.5$ for secant method). *Number of iterations and approximations of p for both methods shown in the command window accompanying code.*

```
% Week 4 code 151A
% Find root of function f(x)=cos(x)-x^3 using Newton's method and Secant method
clc;
clear all;
%% Parameters
tol = 1e-5; % error tolerance
N0 = 500; % maximum number of iterations
p0 = 1; % starting point
f = @(x) exp(x)+2^(-x)+2*cos(x)-6;
p_root=1.82938360193385;
%% Newton's method
f_diff = @(x) exp(x)-log(2)*2^(-x)-2*sin(x);
[iter, p] = Newton(f, f_diff, tol, N0, p0,p_root);
fprintf('Newton''s method:\n')
fprintf('Iteration number = %d \n', iter);
fprintf('p = %.11f \n',p);
fprintf('f(p) = %.11f \n\n', f(p));
%% Secant method
p1 = 1.5;
[iter, p] = Secant(f, tol, N0, p0, p1,p_root);
fprintf('Secant method:\n')
fprintf('Iteration number = %d \n', iter);
fprintf('p = %.11f \n',p);
fprintf('f(p) = %.11f \n\n', f(p));
%% Algorithms
function [iter, p] = Newton(f, f_diff, tol, N0, p0,p_root)
j = 1;
p = p0;
while j < N0
    y = f(p);
    y_diff = f_diff(p);
    % always a good idea to add checks
    if abs(y_diff) < 1e-12
        error('dividing by zero')
    end
    p_next = p - y / y_diff;
    if abs(p-p_root)<tol;
        break;
    end
    j = j + 1;
    p = p_next;
end
p = p_next;
iter = j;
end
function [iter, p] = Secant(f, tol, N0, p0, p1,p_root)
j = 1;
p = p1;
p_prev = p0;
while j < N0
    y = f(p);
    y_prev = f(p_prev);
    % always a good idea to add checks
    if abs(y - y_prev) < 1e-12
        error('dividing by zero')
    end
    p_next = p - y * (p - p_prev) / (y - y_prev);
    if abs(p-p_root)<tol;
        break;
    end
    j = j + 1;
    p_prev = p;
    p = p_next;
end
p = p_next;
iter = j;
end
```



```
Newton's method:
Iteration number = 8
p = 1.82938360193
f(p) = 0.00000000000

Secant method:
Iteration number = 7
p = 1.82938360195
f(p) = 0.00000000008

>>
```

**Problem 2: (C) Newton's Method for Optimization**

Use Newton's method to approximate the value of $x$ that produces the point on the graph of $y = x^2$ that is closest to $(1, 0)$. Use the stopping criterion $|p_{n+1} - p_n| \leq 10^{-8}$ and the value $p_0 = 1$. Report the approximation and the number of iterations needed to reach your computed solution.

*Hint: Minimize $d(x)^2$, where $d(x)$ represents the distance from $(x, x^2)$ to $(1, 0)$.*

**Solution:**

$d(x) = \sqrt{x^4 + (x-1)^2}$. Because distance is always non-negative and $x^2$ is monotone, $d(x)$ is minimized when $d(x)^2$ is minimized.
$(d(x)^2)' \in C^2[0,1]$. By observing the graph of $(d(x)^2)'$ there exists $p \in (0,1)$ s.t $(d(p)^2)' = 0$ and $(d(p))'' > 0$, so there exists a sequence $(p_n)_{n=1}^{\infty}$ that converges to $p$ for any initial $p_0$ within some $\delta$ of $p$.

Let $f(x) = x - \frac{d'(x)^2}{d''(x)^2} = x - \frac{4x^3+2x-2}{12x^2+2}$ with initial estimate $p_0 = 1$.
It takes 6 iterations to obtain a $10^{-8}$-accurate approximation with $p_6 = 0.58975451230$.

4/25/23 4:25 PM /Users/theelusivegerbilf.../hw_3_q_2_new.m 1 of 1

```
% Week 4 code 151A
% Find root of function f(x)=cos(x)-x^3 using Newton's method and Secant method
clc;
clear all;
%% Parameters
tol = 1e-8; % error tolerance
N0 = 500; % maximum number of iterations
p0 = 1; % starting point
f = @(x) 4*x^3+2*x-2;
%% Newton's method
f_diff = @(x) 12*x^2+2;
[iter, p] = Newton(f, f_diff, tol, N0, p0);
fprintf('Newton''s method:\n')
fprintf('Iteration number = %d \n', iter);
fprintf('p = %.11f \n',p);
fprintf('d(p) = %.11f \n\n', sqrt((p)^4+(p-1)^2));

%% Algorithms
function [iter, p] = Newton(f, f_diff, tol, N0, p0)
j = 1;
p = p0;
while j < N0
    y = f(p);
    y_diff = f_diff(p);
    % always a good idea to add checks
    if abs(y_diff) < 1e-12
        error('dividing by zero')
    end
    p_next = p - y / y_diff;
    if abs(p_next-p)<tol
        break;
    end
    j = j + 1;
    p = p_next;
end
p = p_next;
iter = j;
end
```

4/25/23 4:26 PM MATLAB Command Window 1 of 1

```
Newton's method:
Iteration number = 6
p = 0.58975451230
d(p) = 0.53784144870

>>
```

**Problem 3: (T) Convergence Rate**

a) Show that the sequence $p_n = 10^{-2^n}$ converges quadratically to zero.

b) Show that for any positive $k > 1$, the sequence $p_n = 10^{-n^k}$ does not converge quadratically to zero.

c) The sequence $p_n = \sqrt{p_{n-1}}$ starting at $p_0 = 1.5$ converges to $p = 1$. Show that it is linearly convergent (without explicitly solving $p_n$ as a function of $n$).

d) The sequence $p_n = p_{n-1} - \frac{1}{5}\, p_{n-1}^5$ starting at $p_0 = 0.5$ converges to $p = 0$. Show that it converges sublinearly (without explicitly solving $p_n$ as a function of $n$).

e) Recall that a sequence $\{p_n\}$ converges *superlinearly* to $p$ if

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|} = 0.$$

Show that if $p_n \to p$ of order $a$ for $a > 1$, then $p_n$ converges superlinearly to $p$.

**Solution:**

a) If $p_n$ converges to $p$ quadratically, it automatically has to converge linearly as well. $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - 0|}{|p_n - 0|^2} = \lim\limits_{n\to\infty} \dfrac{|10^{-2^{n+1}} - 0|}{|10^{-2^n} - 0|^2} = \lim\limits_{n\to\infty} \dfrac{|10^{-2^{n+1}}|}{|10^{-2\cdot 2^n}|} = \lim\limits_{n\to\infty} \dfrac{|10^{-2^{n+1}}|}{|10^{-2^{n+1}}|} = 1$
Therefore, by the definition of order of convergence, $p_n$ converges to 0 with order of convergence 2 and error constant 1. Since the order is 2, $p_n$ converges quadtratically to 0.

b) $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - 0|}{|p_n - 0|^2} = \lim\limits_{n\to\infty} \dfrac{|10^{-(n+1)^k} - 0|}{|10^{-n^k} - 0|^2} = \lim\limits_{n\to\infty} \dfrac{|10^{-(n+1)^k}|}{|10^{-2\cdot n^k}|} = \lim\limits_{n\to\infty} |10^{2\cdot n^k - (n+1)^k}| = \lim\limits_{n\to\infty} |10^{(n+1)^k (2(\frac{n}{n+1})^k - 1)}| = \infty$ because $(n+1)^k$ diverges to infinity while $2(\frac{n}{n+1})^k - 1)$ converges to one, so $(n+1)^k(2(\frac{n}{n+1})^k - 1)$ diverges to infinity$\Rightarrow$ $|10^{(n+1)^k(2(\frac{n}{n+1})^k-1)}|$ diverges to infinity.

c) $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - 1|}{|p_n - 1|} = \lim\limits_{n\to\infty} \dfrac{|\sqrt{p_n} - 1|}{|p_n - 1|} = \lim\limits_{n\to\infty} \dfrac{1}{|\sqrt{p_n} + 1|} = \dfrac{1}{2}$ because $p_n$ converges to 1.
Therefore, by the definition of order of convergence, $p_n$ converges to 1 with order of convergence 1 and error constant $\frac{1}{2}$. Since the order is 1, $p_n$ converges linearly to 1.

d) $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - 0|}{|p_n - 0|} = \lim\limits_{n\to\infty} \dfrac{|p_n - \frac{1}{5}p_n^5|}{|p_n|} = \lim\limits_{n\to\infty} |1 - \dfrac{1}{5} p_n^4| = 1$ because $p_n$ converges to 0.
Therefore, by the definition of order of convergence, $p_n$ converges to 0 with order of convergence 1 and error constant 1. Since the order is 1 and $\lambda \geq 1$, $p_n$ converges sublinearly to 0.

e) If $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - p|}{|p_n - p|^a} = \lambda$ for $\lambda \in \mathbb{R}^+$. It follows $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - p|}{|p_n - p|} = \lambda \lim\limits_{n\to\infty} |p_n - p|^{a-1}$.
Because $p_n \to p$ and $a > 1$, it follows that $\lambda \lim\limits_{n\to\infty} |p_n - p|^{a-1} = 0$. Therefore $\lim\limits_{n\to\infty} \dfrac{|p_{n+1} - p|}{|p_n - p|} = 0$. Hence $p_n \to p$ superlinearly whenever $p_n \to p$ of order $a > 1$.

**Problem 4: (T) Polynomial Approximation**

Let $f(x) = e^{2x}$ for $x \in [0, 2]$. Find the Lagrange interpolating polynomial of degree-2, i.e. $P_2(x)$, using the nodes $x_0 = 0$, $x_1 = 1$, and $x_2 = 2$ and use it to approximate $f(1.5)$ i.e. $f(1.5) \approx P_2(1.5)$.

**Solution:**

$P_2(x) = \frac{(x-1)(x-2)}{2} - e^2(x)(x-2) + \frac{e^4(x)(x-1)}{2}$
$P(1.5) = 25.89110$, $f(1.5) = 20.08554$ Error$= 5.80556$

**Problem 5: (T) Lagrange Interpolating Polynomials**

Consider the function $f(x) = \frac{14}{3}x^{100} - \frac{64}{59}x^{50} - 97$. Find the Lagrange interpolating polynomial of degree-200 on $[-1, 1]$ using equally spaced nodes $x_j = -1 + j\,h$ for $j = 0, \cdots, n$ with $h = \frac{1}{100}$.

*Hint: The solution for this problem should only be one sentence, no computing or derivations are needed.*

**Solution:**

$P(x) = f(x) - \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)(x-x_1)...(x-x_n) = f(x) = \frac{14}{3}x^{100} - \frac{64}{59}x^{50} - 97$ because $f$ is only 101 times differentiable.

More clearly $P(x) = \frac{14}{3}x^{100} - \frac{64}{59}x^{50} - 97$ over the interval.

**Problem 6: (T) Lagrange Interpolating Polynomials**

Let $P_3(x)$ be the degree-3 Lagrange interpolating polynomial using the input-output pairs $(0,0)$, $(0.5,s)$, $(1,3)$, and $(2,2)$. Find the value of $s$ so that the coefficient of the cubic term $x^3$ in $P_3(x)$ is equal to 6.

**Solution:**

$P_3(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x) + f(x_3)L_3(x)$
$f(0) = 0 \Rightarrow f(0)L_0(x) = 0$
$f(0.5)L_1(x) = s\frac{x(x-1)(x-2)}{0.5\cdot(-0.5)\cdot(-1.5)} = \frac{8sx(x-1)(x-2)}{3}$
$f(1)L_2(x) = 3\frac{x(x-0.5)(x-2)}{1\cdot 0.5\cdot(-1)} = -6x(x-0.5)(x-2)$
$f(2)L_3(x) = 2\frac{x(x-0.5)(x-1)}{2\cdot 1.5\cdot 1} = \frac{2x(x-0.5)(x-1)}{3}$
$f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x) + f(x_3)L_3(x)$
$= \frac{8sx(x-1)(x-2)}{3} - 6x(x-0.5)(x-2) + \frac{2x(x-0.5)(x-1)}{3}$
$= \frac{8s-16}{3}x^3 + (14-8s)x^2 + \frac{16s-17}{3}x$
If $s = 4.25$ then $P_3(x) = 6x^3 - 20x^2 + 17x$